

Abstracts

An Anomaly in Spare-Time Characteristics of Certain Programs Running in a Paging Machine, L. A. Belady, R. A. Nelson, and G. S. Shedler, *Communications of the ACM* **12**, No. 6, 349-353 (June 1969). The running time of programs in a paging machine generally increases as the store in which programs are constrained to run decreases. Experiments, however, have revealed cases in which the reverse is true: a decrease in the size of the store is accompanied by a decrease in running time. An informal discussion of the anomalous behavior is given, and for the case of the FIFO replacement algorithm a formal treatment is presented.

A Base for a Mobile Programming System, R. J. Orgass and W. M. Waite (University of Colorado, Boulder, Colorado), *Communications of the ACM* **12**, No. 9, 507-509 (September 1969). An algorithm for a macro processor which has been used as the base of an implementation, by bootstrapping, of processors for programming languages is described. This algorithm can be easily implemented on contemporary computing machines. Experience with programming languages whose implementation is based on this algorithm indicates that such a language can be transferred to a new machine in less than one man-week without using the old machine.

File Organization: On the Selection of Random Access Index Points for Sequential Files, S. P. Ghosh and M. E. Senko, *Journal of the Association for Computing Machinery* **16**, No. 4, 569-579 (October 1969). The construction of a hierarchy of indexes (the indexed sequential access method) is one means of providing rapid random access to sequential files. An examination is made of the consequences of partially or completely replacing one or more index levels by linear interpolation procedures. For all possible configurations of the several types of key distributions investigated, linear interpolation on the average provides significant performance improvements. Typically, the two accesses required to obtain track index and data are reduced to 1.1 to 1.7 accesses per record. Extremely unusual key distribution will, however, raise the number of accesses required above 2.

Methods Used in an Automatic Logic Design Generator (ALERT), T. D. Friedman and S. Yang, *IEEE Transactions on Computers* C-18, No. 7, 593-614 (July 1969). The ALERT system converts preliminary high-level descriptions of computers into logic. The input to ALERT depicts the architecture of a proposed machine in a form of Iverson notation. As output, the architecture is "compiled" into Boolean equations, which may then be converted into standard computer circuits. The purpose is to relieve designers of uncreative detail work. Complex structures may be represented conveniently by macro functions, algorithms, multidimensional arrays, and subscripted expressions. Control logic, intermediate registers, and selection and switching mechanisms are automatically supplied and the resulting design is consolidated and simplified by a variety of techniques. Methods used and reasons for those approaches are discussed. To elucidate operation of the system, a sample design is followed through its gradual development as it is processed by the successive steps of ALERT. Nine pages of circuit diagrams were automatically generated from the example and are included in the Appendix.

A Note on Storage Fragmentation and Program Segmentation, B. Randell, *Communications of the ACM* 12, No. 7, 365-369 (July 1969). The main purpose of this paper is the presentation of some of the results of a series of simulation experiments investigating the phenomenon of storage fragmentation. Two different types of storage fragmentation are distinguished: (1) external fragmentation, namely the loss in storage utilization caused by the inability to make use of all available storage after it has been fragmented into a large number of separate blocks; and (2) internal fragmentation, the loss of utilization caused by rounding up a request for storage, rather than allocating only the exact number of words required. The most striking result is the apparently general rule that rounding up requests for storage, to reduce the number of different sizes of blocks coexisting in storage, causes more loss of storage by increased internal fragmentation than is saved by decreased external fragmentation. Described also are a method of segment allocation and an accompanying technique for segment addressing which take advantage of the above result. Evidence is presented of possible advantages of the method over conventional paging techniques.

Parallel Program Schemata, R. M. Karp (University of California, Berkeley, California) and R. E. Miller, *Journal of Computer and System Sciences*, 3, No. 2, 147-195 (1969). This paper introduces a model called the parallel program schema for the representation and study of programs containing parallel sequencing. The model is related to Ianov's program schema, but extends it, both by modelling memory structure in more detail and by admitting parallel computation. The emphasis is on decision procedures, both for traditional properties, such as equivalence, and for new properties particular to parallel computation, such as determinacy and boundedness.

A Programmable Waveform Generator, J. D. Bagley and H. B. Baskin, *Proceedings of the IEEE* 57, No. 6, 1207-1208 (June 1969). A technique for using a computer-driven CRT display as a programmable waveform generator is described. The desired signal may be specified by a lightpen on the face of the CRT. The signals produced may have a minimum rise time of 7.25 μ s and be repeated at a rate of up to 69 kHz.

Die Programmiersprache RPG (RPG Programming Language), J. S. Ottmann, *Zeitschrift für Datenverarbeitung* 4, 236-241 (June 1969). The Report Program Generator (RPG) is a programming language for commercial applications. The RPG compiler prepares a fixed logic cycle for each program. The programmer completes this logic cycle by declarative statements. This paper is aimed at those who have no previous experience with RPG and is meant as a basic introduction into this language. Included are the following sections: problems which can be solved with RPG; source program, coding sheets, compilation and diagnostics; processing multiple input files (matching and chaining), table look up, assembler-coded subroutines; new RPG features: EXPT, CHAIN, RPG-coded subroutines.

Roster of Programming Language—1969, J. E. Sammet, *Computers and Automation* 18, No. 7, 153-158 (June 30, 1969). This article is basically a list of higher level languages developed or reported in the U. S. which have been implemented on at least one computer and believed to be in use. Approximately 130 languages are listed, along with the meaning of their acronym, a one or two sentence description of the language, the one or two best reference documents, and where relevant a contact person or organization.

Slower Data Flow Schemes for a High-Speed Serial Store, G. G. Langdon, *Computer Design* 8, No. 7, 48-55 (July 1969). This paper describes a method in which a single high-speed serial storage loop may be used in conjunction with slower circuits to form interesting multiple path storage loops of varying lengths.

Errata

The paper "Internal sorting with minimal comparing" by L. J. Woodrum in Volume 8, Number 3 (1969), has the following corrections:

On page 193, the equation $w(n) = (n - 1)w(a) + w(b)$ should read $w(n) = (n - 1) + w(a) + w(b)$.

On page 195, the statement "where $i = \lfloor \log_2 n \rfloor$ " should read "where $i = \lceil \log_2 n \rceil$."

The APL programs shown in Figure 1 on page 197 and Figure 3 on page 199 should be exchanged.